# Plan de acción para mejorar el proceso de ingeniería de software

1. Evaluación inicial, capacitación y seguimiento
    a. Realizar una evaluación inicial de las habilidades y conocimientos de manera individual de los que conforman el equipo de desarrollo en relación con las mejores prácticas, herramientas y metodologías de desarrollo ágil.
    b. Organizar sesiones de capacitación para abordar las áreas identificadas en la evaluación inicial, incluyendo temas como revisión de código, UX/UI, linters y guías de mejores prácticas.
2. Establecer guías de mejores prácticas
    a. Crear un repositorio centralizado de guías y documentos de mejores prácticas para cada lenguaje de programación y framework.
    b. Incluir en las guías información sobre cómo utilizar linters y cómo configurarlos en los diferentes entornos de desarrollo.
    c. Establecer un proceso para actualizar periódicamente las guías de mejores prácticas con base en las retroalimentaciones y aprendizajes del equipo.
3. Implementar un proceso de revisión de código
    a. Establecer un proceso de revisión de código para todos los proyectos, en el cual los desarrolladores revisen y comenten el código de sus colegas antes de que se integre al repositorio principal.
    b. Asignar a los desarrolladores senior el rol de **revisores expertos** para que puedan dar retroalimentación constructiva y ayudar a mejorar la calidad del código.
4. Mejorar la UX/UI
    a. Contratar diseñadores de UX/UI o capacitar a algunos miembros del equipo de desarrolladores en diseño de interacción y experiencia de usuario.

b. Integrar a los diseñadores de UX/UI en los equipos de desarrollo para que trabajen en colaboración con los desarrolladores desde el inicio y durante el desarrollo de proyectos.

5. Mejorar la toma de decisiones y la comunicación
    a. Establecer reuniones semanales de Scrum para que los equipos compartan sus avances y retos, permitiendo una comunicación más fluida y la identificación temprana de problemas.
    b. Fomentar la toma de decisiones, motivando a los equipos para que tomen decisiones sobre sus proyectos dentro de los limites permisibles.
    c. Implementar herramientas de comunicación interna como Slack o Microsoft Teams para facilitar la colaboración y el intercambio de conocimientos entre los miembros del equipo.
6. Optimizar la retroalimentación con los clientes
    a. Establecer reuniones regulares con los clientes para discutir el progreso del proyecto y recibir retroalimentación sobre las entregas.
    b. Implementar un sistema de seguimiento de problemas para que los clientes puedan reportar errores y solicitar mejoras de manera más eficiente.
7. Fomentar la colaboración y el acceso a recursos externos
    a. Establecer sesiones periódicas de programación en pareja o mob programming, donde los desarrolladores trabajen juntos en la resolución de problemas y compartan sus conocimientos.
    b. Fomentar la participación de los desarrolladores en conferencias, talleres y cursos en línea para mantenerse actualizados en las últimas tendencias y tecnologías.
8. Seguimiento y ajuste
    a. Realizar evaluaciones periódicas del progreso y los resultados de las mejoras implementadas, ajustando el enfoque según sea necesario para garantizar el éxito del plan.
    b. Establecer un sistema para monitorear la productividad de los desarrolladores, la calidad del software, la satisfacción de los clientes y la integración del equipo. Utilizar estos sistemas para identificar áreas de mejora y ajustar las estrategias.
9. Generación de propuestas y crecimiento de la empresa

a. Establecer un proceso para la generación de propuestas que incluya la identificación de oportunidades de negocio, la preparación de propuestas técnicas y comerciales y la presentación de las mismas a los clientes potenciales.

b. Involucrar a los desarrolladores en la generación de propuestas, aprovechando sus conocimientos técnicos y experiencia en el desarrollo de soluciones similares.

c. **Fomentar la innovación y la experimentación** en la empresa, creando un entorno que permita a los desarrolladores proponer nuevas ideas y soluciones que puedan convertirse en futuros proyectos o servicios.

10. Mentorías y programas de desarrollo profesional

    a. Establecer un programa de mentoría en el que los desarrolladores senior apoyen a los desarrolladores de nivel medio en su crecimiento profesional y técnico.

    b. Crear planes de desarrollo profesional individuales para cada desarrollador, identificando áreas de mejora, objetivos y oportunidades de capacitación.

    c. Fomentar la participación de los desarrolladores en otras comunidades de desarrolladores y grupos de interés, facilitando la colaboración y el intercambio de conocimientos con profesionales de otras organizaciones.

11. Fomentar la retroalimentación y reconocimiento

    a. Implementar un sistema de retroalimentación entre el equipo de desarrolladores, donde puedan ofrecer comentarios constructivos con referencia a las mejores prácticas.

    b. Reconocer y celebrar los logros individuales y del equipo, fomentando una cultura de aprecio y motivación en la empresa.

12. Adoptar metodologías ágiles y adaptativas:

    a. Asegurar que los equipos de desarrollo apliquen de manera efectiva las prácticas ágiles de Scrum, ajustándolas según las necesidades específicas de cada proyecto y cliente.

    b. Promover la adopción de metodologías y prácticas de DevOps para optimizar la entrega y el mantenimiento del software.

13. Gestionar y mitigar riesgos

a. Implementar un proceso de identificación y gestión de riesgos en los proyectos, asegurando que los equipos estén preparados para abordar los desafíos y problemas que puedan surgir; antes, durante y posterior al desarrollo de los proyectos.

b. Establecer mecanismos de comunicación y atención de riesgos, facilitando la toma de decisiones y la asignación de recursos para resolver problemas críticos.

c. Mitigar la falta de compromiso y responsabilidad de los desarrolladores en el trabajo remoto, mediante las siguientes estrategias:

    i. Establecer metas claras y plazos específicos para cada tarea asignada, asegurando que los desarrolladores comprendan sus responsabilidades y expectativas de desempeño.

    ii. Implementar un sistema de seguimiento y revisión del trabajo, donde los desarrolladores informen regularmente sobre su progreso y reciban retroalimentación de sus compañeros.

    iii. Fomentar la comunicación y colaboración entre los miembros del equipo mediante herramientas como Slack o Microsoft Teams, permitiendo la discusión abierta de problemas y soluciones, así como el apoyo mutuo en el trabajo.

    iv. Establecer una política de trabajo remoto que incluya expectativas de disponibilidad, comunicación y rendimiento, y proporcionar orientación y recursos para ayudar a los desarrolladores a cumplir con estos requisitos.

    v. Realizar reuniones periódicas de seguimiento con los desarrolladores para discutir su progreso, abordar problemas o inquietudes, y ofrecer apoyo y orientación en el cumplimiento de sus responsabilidades.

14. Mantenimiento y mejora continua

    a. Establecer un proceso de mejora continua en la empresa, revisando y ajustando periódicamente el plan de acción en función de los resultados obtenidos y las necesidades del negocio.

    b. Promover la adopción de nuevas tecnologías y metodologías de desarrollo que puedan ayudar a mejorar la productividad de los desarrolladores, la calidad del software y la satisfacción de los clientes.

c. Establecer una cultura de aprendizaje y colaboración en la empresa, en la que los desarrolladores se sientan motivados para seguir mejorando sus habilidades y compartir sus conocimientos con los demás miembros del equipo.

15. Evaluación y ajuste continuo
    a. Realizar evaluaciones trimestrales del progreso y los resultados del plan de acción, ajustando las estrategias y prioridades según sea necesario para garantizar el éxito a largo plazo.
    b. Fomentar una cultura de adaptación y aprendizaje en la empresa, asegurando que los desarrolladores y equipos estén abiertos al cambio y dispuestos a ajustar sus prácticas y enfoques en función de las necesidades del negocio y los clientes.
16. Crear equipos multidisciplinarios
    a. Formar equipos de desarrollo que incluyan desarrolladores, diseñadores de UX/UI, analistas de negocio, arquitectos de software y administradores de proyectos para asegurar una perspectiva integral en cada proyecto.
    b. Promover la colaboración y el intercambio de conocimientos entre los diferentes roles, permitiendo que cada miembro del equipo contribuya con sus habilidades y experiencia al éxito del proyecto.
17. Implementar herramientas de gestión de proyectos
    a. Adoptar herramientas de gestión de proyectos y seguimiento del trabajo, como Jira o GitHub Boards, para mejorar la planificación, seguimiento y control de los proyectos.
    b. Capacitar a los miembros del equipo en el uso eficiente de estas herramientas y asegurar que todos sigan las mismas prácticas y metodologías de trabajo.
18. Fomentar la automatización y eficiencia
    a. Implementar procesos de automatización en el desarrollo y despliegue del software, utilizando herramientas como Jenkins, GitLab CI/CD o GitHub Actions, para mejorar la velocidad y calidad de las entregas.
    b. Identificar áreas en las que la automatización pueda reducir la carga de trabajo y los errores, permitiendo a los desarrolladores enfocarse en tareas de mayor valor.

c. Integrar soluciones de inteligencia artificial como GitHub Copilot para mejorar la productividad del equipo de desarrollo. Estas herramientas pueden asistir en la generación de código, sugerir soluciones a problemas comunes y proporcionar información útil basada en la experiencia de otros desarrolladores. Capacitar a los miembros del equipo en el uso eficiente de estas tecnologías y fomentar su adopción en el flujo de trabajo diario.

19. Establecer alianzas con otras empresas y proveedores
    a. Desarrollar alianzas estratégicas con otras empresas y proveedores de tecnología para expandir la oferta de servicios y soluciones, así como mejorar el acceso a recursos y conocimientos externos.
    b. Participar en eventos de la industria y colaborar con otras organizaciones en proyectos y iniciativas conjuntas, fomentando el intercambio de ideas y la innovación.
20. Medición y seguimiento del éxito
    a. Continuar el seguimiento de las herramientas establecidas y ajustarlas según sea necesario para reflejar las prioridades y objetivos cambiantes del negocio.
    b. Realizar encuestas de satisfacción y retroalimentación con los clientes y miembros del equipo, utilizando los resultados para identificar áreas de mejora y ajustar el enfoque del plan de acción.
21. Establecer un modelo de trabajo basado en DevOps e Ingeniería Ágil:
    a. Integrar las prácticas de DevOps en los equipos de desarrollo y operaciones, fomentando la colaboración y la comunicación constante entre ambas áreas.
    b. Adoptar metodologías ágiles en la gestión de proyectos y en el desarrollo del software, asegurando la flexibilidad y adaptabilidad ante cambios en los requisitos y las necesidades del cliente.
    c. Implementar herramientas y tecnologías para facilitar la adopción del modelo DevOps, como contenedores (Docker), infraestructura como código (Terraform) y herramientas de monitoreo y seguimiento de aplicaciones (Prometheus, Grafana).
22. Robustecer nuestra operación con mejores prácticas en Software:

a. Adoptar estándares de calidad como ISO15504 para evaluar y mejorar la madurez del proceso de desarrollo de software.

b. Capacitar a los desarrolladores en principios y patrones de diseño como SOLID, Singleton Pattern y diferentes arquitecturas (Clean, MVVM, MVC, VIPER, etc.), para garantizar la creación de código estructurado, mantenible y escalable.

23. Mejorar el seguimiento a proyectos a través de agilidad (Control de HxH):

    a. Implementar el control de horas-hombre (HxH) para monitorear la asignación y utilización de recursos en los proyectos.

    b. Utilizar herramientas de seguimiento de tiempo y reporte de actividades para mejorar la visibilidad y transparencia en la asignación y uso de recursos.

    c. Analizar los datos de HxH para identificar áreas de mejora en la planificación y ejecución de proyectos, y ajustar las estrategias y prioridades según sea necesario.

24. Automatizar pruebas dentro de los proyectos:

    a. Implementar pruebas automatizadas en el proceso de desarrollo del software, utilizando herramientas como Jenkins, Selenium, JUnit o XCTest, para garantizar la calidad y reducir errores en el código.

    b. Capacitar a los desarrolladores en la creación de pruebas unitarias, de integración y de aceptación para validar y verificar el correcto funcionamiento del software.

    c. Establecer métricas y objetivos de cobertura de pruebas para asegurar que las pruebas automatizadas cubran un porcentaje adecuado del código y de los casos de uso.

25. Cubrir proyectos JIT, mejorar la productividad de nuestros equipos y especializar el entorno tecnológico de XID:

    a. Establecer un proceso de asignación de proyectos JIT (Just-In-Time) que permita asignar recursos de manera eficiente y oportuna a los proyectos que lo requieran.

    b. Implementar prácticas de trabajo y herramientas que mejoren la productividad de los equipos de desarrollo, como revisiones de código, linters, automatización de tareas y acceso a recursos externos.

c. Fomentar la especialización en el entorno tecnológico de XID, capacitando a los desarrolladores en tecnologías específicas, herramientas y mejores prácticas para fortalecer las capacidades del equipo.

26. Identificar capacidades del equipo:

    a. Realizar una evaluación de las habilidades y competencias de los miembros del equipo, identificando sus áreas de fortaleza y oportunidades de mejora.

    b. Crear un inventario de habilidades y conocimientos de los miembros del equipo, lo que permitirá asignar de manera eficiente los recursos a proyectos y tareas que se ajusten a sus capacidades y experiencia.

    c. Establecer un programa de capacitación y desarrollo basado en las necesidades identificadas durante la evaluación, proporcionando oportunidades de aprendizaje y crecimiento a los miembros del equipo.

    d. Fomentar la mentoría y el apoyo entre los miembros del equipo, incentivando a los desarrolladores senior a compartir su experiencia y conocimientos con los miembros menos experimentados.

    e. Establecer un proceso de revisión periódica de las capacidades del equipo, con el fin de monitorear y ajustar el enfoque de desarrollo y capacitación según las necesidades cambiantes del negocio y la evolución del mercado tecnológico.

Al seguir expandiendo y ajustando el plan de acción en respuesta a los resultados obtenidos, las necesidades del negocio y los cambios en la industria, se logrará una mejora continua en el proceso de ingeniería de software y en la empresa en general. Esto permitirá enfrentar eficazmente los desafíos actuales y futuros, garantizando el éxito a largo plazo y la satisfacción de los clientes y desarrolladores.